대한산업공학회/한국경영과학회 2000 춘계공동학술대회 논문집. 2000년 4월 21-22일. 경남대학교
Session 0C7.3

설계변경 관리를 위한 제품 자료 표현
A Product Data Representation for Engineering Change Management

도남철
볼보건설기계코리아(주)

Abstract

This paper provides a product data representation, a data schema and related operations, for the management of engineering changes. The computer interpretable format of the representation enables itself to be realized through databases and their applications to a component of a product data management system. Supporting general operations for the engineering changes, it resolves problems of the existing information systems in the nested engineering changes and the simultaneous change application to multiple products. To show the feasibility, a prototype system is implemented based on the representation.

1. 서론

제품을 위한 자료표현(Product data representation)은 제품 수명주기동안에 발생하는 정보를 처리하기 위한 자료구조(Data structure)와 관련 연산(Operations)을 표현하는 모델이다. 이 자료표현을 기반으로 다양한 정보 시스템들이 정보를 관리하거나 상호 교환하고 있다. 그러므로 효과적인 자료표현은 제품개발, 생산, 그리고 고객지원 과정에서 일관적이고 통합된 정보 지원이 가능하게 한다.

자료표현의 예로 생산공장에서 생산에 필요한 자원을 계획하는데 필요한 Bills of Material(BOM)을 들 수 있다. 최종생산 물량에서 필요한 부품 및 생산 자원을 계산하기 위하여 BOM은 제품을 부품과 필요수량을 포함한 부품간의 조립관계로 표현하고 있다.

설계변경은 부품정보를 변화시킬 뿐 만 아니라 제품의 구조(부품간의 조립관계)를 변화 시킨다. 제품의 구조 변화는 한 모델 안에서 새로운 사양(Configuration)을 생성시킨다. 아울러 설계변경은 변경된 부품을 포함하는 모든 모델 및 사양에 대하여 변경을 적용할 지에 대한 의사결정을 유발시킨다. 그러므로 설계변경을 지원하는 자료표현은 다양하고 변화하는 설계변경의 요구를 표현할 수 있어야 한다.

최근의 Network 화 된 산업환경에서는 설계변경은 공급 혹은 수요자 망을 통하여 회사 밖으로 빠르게 전달되어야 한다. 특히 다국화 된 기업에서는 광속의 제품변경 정보의 유통이 경쟁력의 한 요소이다. 광범위하고 빠른 설계변경 정보의 유통을 위하여 모든 설계정보는 정보시스템 내에서 효과적으로 저장, 관리되어야 하며, 이를 지원하기 위하여 통합되고 표현력이 뛰어난 제품자료 표현이 필요하게 된다.

이 연구의 목적은 설계변경을 효과적으로 지원하는 제품자료표현을 제안하는 것이다. 이 표현을 제공하기 위하여 전형적인 예를 도입하고 이 예로부터 필요한 기능을 도출한다. 도출된 기능을 지원하기 위한 형식화된 자료구조와 자료 처리를 정의하며 이 정의는 Computer 를 이용한 자료처리(Database 를 이용한 관리)가 가능한 형태로 표현된다. 유효성을 보이기 위하여 제안된 자료표현을 기반으로 시제품을 개발하였다.

2. 관련연구

제품자료 교환의 국제적 표준인 ISO STEP PART 41, 44, 203[1,2,3]의 configurable item 과 effectivity 개념을 제안한 제품자료표현의 기본 개념으로 수용하였다. 자료구조 관점에서 보면 Tree 구조에서 Labeled Edge 개념[4]을 수용하였으며 Label 이 다른 Edge 가 될 수 있는 확장된 구조가 사용되었다.

[5]에서는 설계변경 시 중복설계변경과 변

경이력 추적에 문제가 있으며 아직 이를 적절히 지원하는 Product Information System 이 없음을 언급하였다.

CATIA CDM[6]은 Part 사이의 Assembly relationship 도 하나의 부속 Object 로 보아 Locking 을 처리하였으나, 설계변경 시 같은 Hierarchy 구조상의 Part 에 대해서는 Locking Mechanism 이 제공되지 않는다.

Enovia VPM 1.1[7]은 Action Flow 라는 개념으로 설계변경의 이력을 기록하고 이에 일관되게 Effectivity 를 줄 수 있는 Mechanism 을 소개하고 있다. 하지만 Multi Model Application 에 대한 일괄 설계변경 적용만이 지원된다.

기존 Database 연구 및 문서관리 제품[8,9]에서 설계변경관리의 대안으로 Version Model 이나 문서관리를 제시하였으나 이는 부품 자체의 변화를 중점으로 하였고 제품의 구조에 대한 고려가 없었다. 설계변경은 부품내용의 변화에 따른 부품 구조의 변화를 고려해야 한다.

3. 설계변경의 예

그림 1 은 설계변경의 전형적인 예로 1650100 --- 의 1650200 ---로 변경이 제품구조정보와 사양에 미치는 영향을 보여준다. 그림에서는 실선으로 변경 전의 구조를 보여주고 점선으로 변경 후의 구조를 보여준다.

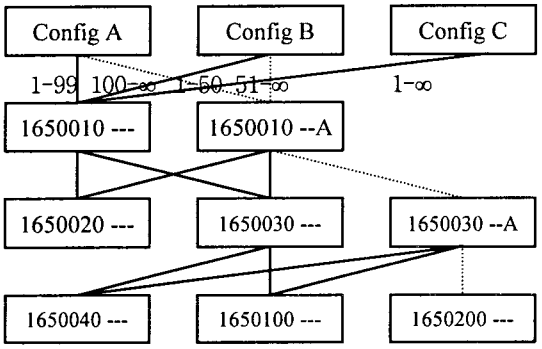

그림 1. 설계변경의 예

부품의 변경은 상위 조립품의 변화를 가져오므로 최상부 조립품에서 제품구조를 변경 시키게 된다. 다음에는 각 부품에 적용된 Operation 을 나열하고 이를 설명한다.

- Replace 1650010 --- with 1650010 --A

여기서 1650010 --A 는 1650010 ---를 Version Up 한 부품이다. Version Up 은 기존의 하위부품연결을 그대로 유지한 체 같은 부품번호를 가지고 한단계 높은 Version 항목 (예에서는 --A) 을 가진 부품을 생성시킨다.

- Replace 1650030 ---
with 1650030 --A (under 1650010 --A)
- Replace 1650100 ---
with 1650200 --- (under 1650030 --A)

위 3 가지 Replace 로 조립품의 설계변경이 완료되었다. 이제는 변경된 조립품을 Configurable 에 적용 여부를 결정하게 된다. 여기서 Configurable 은 제품의 구조가 특정 조건에 따라 서로 다르게 결정될 수 있는 가상의 Part 로써 특정 조건을 Effectivity 라고 한다[3]. 예에서는 3 개의 Configurable A, B 그리고 C 가 있으며 이 중 C 는 생산 중단된 사양이므로 A 와 B 에만 설계변경을 적용하기로 결정하였다. 다음이 해당 Operation 이다.

- Replace 1650010 --- with 1650010 --A
(under Config A)
- Replace 1650010 --- with 1650010 --A
(under Config B)

이제 Configurable A, B 가 조건에 따라 적절한 구조를 가지게 하기 위하여 Effectivity 를 선언하게 된다. 예에서는 serial_numbered_effectivty [1]을 채용하여 A 사양은 100 호기에서부터 설계변경을 적용하여 Effectivity 을 1-99 와 100-∞ 을 정의하고 B 는 1-50 과 51-∞ 을 정의한다.

위의 모든 설계변경 과정은 기록되어야 한다. 이는 설계변경을 검토, 승인 그리고 적용하는데 근간이 되므로 설계변경을 위한 제품정보 표현에 중요한 요소이다. 다음은 제품구조 변경에 대한 설계변경 이력이다.

표 1 설계변경이력

No	part	qty	opt	comp.
1	1650010 ---	1	Replace	y
	1650010 --A	1		
2	1650030 ---	1	Replace	y
	1650030 --A	1		
3	1650100 ---	1	Replace	n
	1650200 ---	1		

이러한 변경 중에 많은 제약조건이 있을 수 있으나 본 논문에서는 두 가지만 고려한다. 첫째 설계변경중인 부품을 다시 설계변경 하는 것 (Nested engineering changes)을 방지하는 기능이다. 이를 위해서는 부품자체와 부품사이의 포함관계에도 변경 방지 장치가 준비되어야 한다. 다음으로는 변경된 부품의 다수 모델에 대한 적용 (Simultaneous application to multiple models) 이다. 위의 예에서 설계 변경된 부품 1650010 --A 는 이를 포함하고 있던 모든 Configurable 에 동시 자동 적용될 수 있어야 하며 시스템은 자동으로 생산종료 된 C 에 대해서는 적용금지 해야 한다 (물론 사용자가 시스템의 경고를 무시하고 임의로 C 에 적용할 수도 있다).

이상으로 우리는 설계변경을 위한 기본적인 요구사항과 특정한 두 가지 제약조건을 예를 통하여 알아보았다. 이를 정리하면:

부품과 부품의 조립관계를 표현

조건에 의하여 하위제품 구조를 가변적으로 정의하는 Configurable
설계변경을 지원하는 연산 (예 Replace)
설계변경 이력의 기록
중복된 설계변경의 방지
설계변경의 동시 다수 Configurable 에 적용 및 의사결정지원

4. 설계변경을 위한 추상적 자료구조

이 장에서는 앞장의 요구사항을 만족시키는 제품자료구조를 정의한다. 특히 이 정의의 자료구조부분은 기존의 Database 에서 사용하는 자료정의언어(Data definition language)로 호환가능 하도록 정의하여 제품자료관리 시스템(Product data management system)의 한 요소로써 사용가능 하도록 고려하였다. 이 정의에서는 객체지향적 표현방법을 사용하여 상속성(Inheritance)과 자료와 연산의 통합 표현(Method)을 채용하였다[10]. 그림 2 는 자료구조의 모습을 보여준다.

4.1 Part 와 Configurable

부품과 부품사이의 조립관계를 나타내는 Entity 는 PART 와 EDGE 이다. PART 의 working_flag 와 EDGE 의 replaced 항목은 각기 Locking 과 설계변경 기록을 위하여 사용된다. CONFIG 와 CONFIG_EDGE 는 Effectivity 에 따라 하부 부품 구조가 변경되는 사양을 표시하며, 각기 PART 와 EDGE 에 상속관계를 가지는 하위 Class 이다.

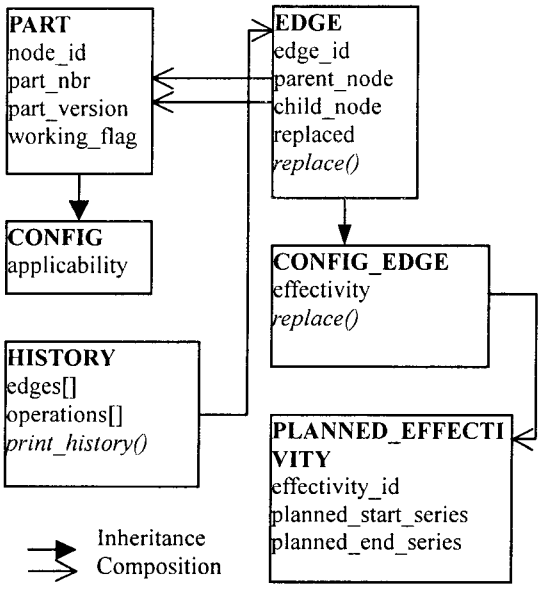

그림 2. 제안된 제품자료구조

4.2 Replace

제품구조(Product Structure)를 변경시킬 수 있는 연산으로써는 Add, Delete, Replace 가 있다. 이들은 모두 EDGE entity 에 적용되는 연산이며 이중 Replace 의 표현을 보면 다음과 같다.

```
EDGE EDGE.replace(PART new)
{ new_id = NEW ID();
pid = self.get_parent_node();
cid = new.get_node_id();
old_edge = self.get_edge_id();
return (NEW EDGE(new_id,pid, cid,old_edge);}

CONFIG_EDGE
CONFIG_EDGE.Replace(CONFIG new)
{ super.Replace();
effectivity = get_eff();}
```

4.3 설계변경 이력

설계변경이력을 위해서 HISTORY entity 와 EDGE 의 replaced 항목이 사용된다. HISTORY 에는 변경된 EDGE 정보와 해당 Operation 이 기록된다. 다음의 print_history method 는 설계변경 이력에서 변경이전과 이후 부품을 자료구조에서 Reference 하는 Path 를 보여준다.

```
HISTORY.print_history()
{ for (i=0; i< Cardinality(self.edges[]); i++)
  { print(self.edges[i].child_node.part_nbr);
  print(self.edges[i].replaced.child_node.part_nbr);}
}
```

4.4 중복설계변경 방지

중복 설계변경 방지를 위해서는 PART 의 working flag 가 사용된다. 제안된 설계변경 모델에서는 설계변경 된 부품에서 최상위 조립품까지가 설계변경의 단위이므로 한 부품이 설계변경 될 경우 해당부품이 포함된 전체조립 참가품에 Locking 을 걸어놓는다. 다음은 working_flag 를 이용한 Locking 정의이다. 이를 표현하기 위하여 Event-Condition-Action 제약조건인 Constraint 를 사용한다[11].

```
CONSTRAINT on EDGE.replace()
IF obj.parent_node.working_flag = TRUE
INVALIDATE TRANSACTION;
```

다음은 Replace 시 하위 제품구조에 포함된 모든 PART 의 working_flag 에 값을 TRUE 로 바꾸는 Method 이다. Replace 시 이 Method 를 해당 변경이 일어나는 부품구조의 Root Part 에 적용한다.

```
PART.turn_on()
{self.working_flag = TRUE;
        WHILE (obj is a EDGE &
         obj.get_parent_node() = self)
        { obj.get_child_node().turn_on()}
}
```

4.5 다중모델적용

다중 모델적용을 위해서는 CONFIG 의 applicability 항목과 다음의 CONSTRAINT 및 Method 을 사용한다.

```
CONSTRAINT ON CONFIG_EDGE.replace()
```

```
EXECUTE obj.multi_app();
```

CONFIG_EDGE 에 Replace Operation 이 일어날 경우 multi_app() Method 가 실행된다.

```
CONFIG_EDGE.multi_app()
{WHILE( edge is a CONFIG_EDGE &
edge.get_child_node() =self.child_node)
{IF edge.get_parent_node().applicablity == TRUE;
NEW config_edge(NEW ID,
edge.get_parent_node(),self.child_node,
edge.get_id());}
}
```

이 Method 는 변경된 PART (self.child_node)를 child_node 로 가지는 모든 CONFIG_EDGE 를 찾아 parent_node 의 Applicability 가 True 인 경우만 (현재 생산중인 경우만) 새로운 CONFIG_EDGE 를 만들어 주게 된다.

5. 구현

제안된 자료표현은 시제품을 통하여 실제 구현 중이다. 그림은 개발중인 시제품의 제품구조 전개 및 부품정보 검색 모습이다.

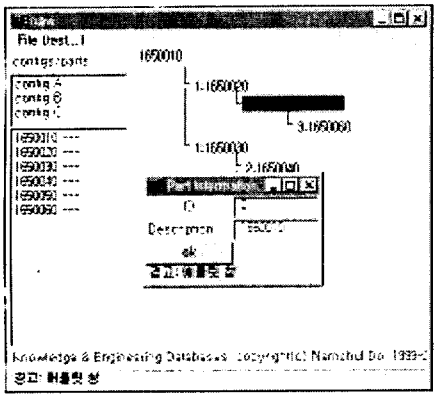

그림 3. 개발중인 시제품의 모습

6. 결론

본 연구에서 설계변경을 위한 제품자료표현 모델을 제시하였다. 이 모델은 제품의 기본적 구조, 사양표현, 설계변경 기록, 중복된 설계변경 및 다중설계변경을 지원한다. 추후 연구 과제로써는 설계변경이 설계에서 생산, 판매, 고객지원 부분으로 전달되면서 발생하는 문제의 해결이다. 이를 해결하기 위하여 설계변경시의 호환성 표현과 설계변경 진행중의 설계변경의 Grouping 기능을 가지는 제품자료표현이 요구된다.

참고문헌

[1] ISO 10303-41, " PART 41: Integrated Resources: Fundamentals of Product Description and Support", ISO, 1994.
[2] ISO 10303-44, " PART 44: Integrated Resources: Product Structure Configuration", ISO, 1994.
[3] ISO 10303-203, " PART 203: Application Protocol: Configuration Controlled Design",ISO, 1994.
[4] Aho, A., Hopcroft, J. and Ullman, J., " Data Structures and Algorithms", Addision-Wesley, 1982.
[5] D.H.Brown Associates, Inc." Product Vs Process in Design-To-Manufacturing Integration", D.H.Brown Report, 1997.
[6] Dassault Systems, " CATIA.Data Management - Access Planning and Administration Guide v4", Dassault Systems, 1997.
[7] Enovia, " Enovia VPM 1.1 User's Guide", Enovia, 1998.
[8] Agilesoft, "Why Document Management Systems Don't Work for the ECO Process", White paper, 1999.
[9] Biliris, A.," Modeling Design Object Relationships in Pegasus", Proc. IEEE Data Engineering Conference, 1990.
[10]김원, 객체지향 데이타베이스 Introduction to Object-Oriented Databases, 하이테크정보, 1994.
[11]도남철, " Backward Propagation of User Defined Integrity Constraints in An Active Object-Oriented Database", 포항공과대학교 산업공학과 박사학위 논문, 1996.